Abstract #: 015-0678

Meta-modeling of An Emergency Department DES Model:

A Neural Networks Approach

Samuel Xu

Operations Management

Haskayne School of Business

University of Calgary

2500 University Dr NW

Calgary, Alberta, Canada T2N 1N4

Email: Samuel.xu@haskayne.ucalgary.ca

Tel: 1-403-274-9708

POMS 21st Annual Conference

Vancouver, Canada

May 7 to May 10, 2010

Meta-modeling of An Emergency Department DES Model:

A Neural Networks Approach

Abstract

In this paper, the author explores the possible use of neural networks to model the generalized response of input changes in a discrete even simulation (DES) model without the need to rerun the model. The DES model was built to diagnose causes of patient delays in the emergency department (ED) at Foothills Medical Center (FMC) in Calgary, Alberta, Canada. The outputs of the DES model are used to train neural networks. Performance comparison is conducted among the trained neural networks, the traditional multivariate regression models, and the DES model itself. The initial result is very encouraging. In the paper, the author also discusses different training strategies for neural networks, and the potential applications of the trained neural networks.

Keywords

Health care, discrete event simulation, neural networks, meta-modeling, and multivariate regression model

1. Introduction

Healthcare is a human-based system, which is highly intolerant to failures. A healthcare system involves multiple stakeholders interacting with each other, and thus is complex in nature. Healthcare providers require tools to comprehend the complex interactions of healthcare and foresee the consequences of their decisions. Computer simulation is one of the most powerful tools to assist healthcare managers in comparing the different consequences of alternative decision scenarios, and has been widely used for healthcare management [1-3]. However, computer simulation is still facing many challenges in healthcare to realize its full potential [4-5]. One of the challenges is that it needs specialized expertise to develop and run a simulation model. Many healthcare organizations do not have this expertise in house, and have to rely heavily on external resources. As a result, computer simulation is often the last tool considered by healthcare managers. How can this situation be improved? The solution proposed in this paper is to use meta-models to reproduce the simulation over wide ranges of interest without the need to rerun the simulation models. A simulation metal model can be easily integrated into other decision supports systems and provide much more efficiency than the simulation itself [6].

In this paper, the author presents a neural network based meta-model for an emergency department (ED) discrete-event simulation (DES). The DES model is run to generate outputs for different input configurations. These outputs are used to train the neural network, which is then used to predict the response of a given input configuration. Analysis of the outputs of the neural network model shows that the trained neural network is able to predict simulation outputs with very good accuracy for input configurations not in the training set, and outperforms multiple regression models developed from the same training set. The study reported in this paper also explores the possible impacts of different neural network configurations on the performance of the neural networks.

2. Overview of the FMC ED Simulation

The emergency department (ED) at the Foothills Medical Centre (FMC) in Calgary, Alberta is an emergency care facility in the Calgary Health Region (CHR) handling over 70,000 patient visits annually. The FMC ED is staffed with 10 emergency physician (EP) shifts per day. It has a total of 53 treatment beds, and 21 to 23 treatment nurses on duty depending on the time of the day. The objective of the FMC ED simulation project was to develop a DES model to identify patient flow problems in the ED and provide a test environment for suggested improvement alternatives.

It took a year and half to develop the DES model for the FMC ED. Major modeling activities include analyzing the ED procedure documents, interviewing various ED staff, analyzing over 30,000 patient records, 37,000 DI (Digital Image) records and 140,000 lab test records, and "shadowing" 20 EP shifts for times spent for various EP activities. The model also incorporates a neural network model to simulate patients' LWBS (leave without being seen) behavior in the waiting room.

The flow of patients through the FMC ED, represented in the DES model, can be described as follows:

- Patients arrive and wait for triage.
- Patients are triaged to determine their acuity score and the type of ED bed required.
- Patients wait for ED beds or LWBS.
- Patients getting ED beds wait for ED nurse assessment.
- ED nurse assessment determines a patient's need for DI or lab tests.
- Patients who need DI or lab tests take those tests and wait for the results to be made available.
- Patients who do not need DI or lab tests and those with their DI or lab test results back wait for a first EP assessment. Some patients may LWBS during this delay also.
- First EP assessment resulting in a disposition decision of one of three types: discharge, take more lab tests, or recommended for admission.

- Patients who need more DI or lab tests take those tests and wait for the results to be made available. After their test results are back, they wait for EP reassessment
- EP reassessment resulting in a disposition decision of one of the three types noted earlier. Patients who need more DI or lab tests will repeat the previous step.
- Patients with admission recommended wait for an assessment from a consulting physician.
- Consulting physician assessment results in a decision to either discharge or admit the patient.
- Patients admitted wait in the ED until an inpatient bed is available in the main hospital.

The DES model was built using ARENA 12, and was validated by comparing the simulation model's output values to those from the actual system. The key performance summary of the DES model is shown in Table 1.

Table 1: Performance Summary of the DES Model

Performance Indicators	Unit	Historical Data	Simulation Result	
			95% CI Low	95% CI High
Wait For 1st EP Assess	Minute	182.7	178.2	185.6
LoS Total	Minute	458.7	456.7	464.2
LoS Admitted	Minute	754.0	751.2	761.8
LoS Discharge	Minute	355.6	350.7	357.9

From the table above, we can see that on every measure, the model results are very close to actual system performance. This model can be used to evaluate the impacts of a variety of process and/or policy changes on the ED performance.

Then the question is: Why do we bother spending time again building a simulation meta-model? This question can be answered from different angles. First, although the ED management is impressed by the power of the DES model, they do not have the necessary knowledge to run the model and interpret the model outputs by themselves. A simulation meta-model can be implemented in EXCEL or other software tools, which most of the people are familiar with. The second one is related to the long time to run the simulation model. It takes about 40 minutes to generate results for each scenario with ten (10)

485-day replications. The time needed for the simulation meta-model to reproduce the results is only a couple of milliseconds. The third one comes from a real need for a simulation meta-model. While working on the DES model, the author was also participating in the development of a system dynamic (SD) model to simulate the patient-flow within the CHR. The SD model was built using Vensim® for Windows from Ventata Systems Inc., and the FMC ED is included in the SD model as a component. As Vensim and ARENA do not talk to each other, so we cannot get the FMC DES model communicate directly with the SD model. Developing a meta-model for the DES model appears to be the best option to link these two models together.

3. Artificial Neural Networks

An artificial neural network, usually called "neural network", is a mathematical system that mimics the way in which the brain works. It consists of several highly interconnected computational elements, known as nodes, neurons, or perceptrons. Each node does the following tasks [7]:

1) Signals are received from other nodes (X_0, X_1, …,X_n)
2) Signals are multiplied by their corresponding weights (W_0X_0, W_1X_1, …,W_nX_n)
3) Weighted signals are summed (Sum = $W_0X_0 + W_1X_1 +…+ W_nX_n$)
4) The calculated sum is transformed by an activation function [F(Sum)]
5) The transformed sum is sent to other nodes (repeats 1-4 above)

Some of the common activation functions are as follows [8-9]:

- Linear or identify: $F(x) = x$
- Hyperbolic tangent: $F(x) = \tanh(x)$
- Threshold: F(x) 0 if x<0, 1 otherwise
- Gaussian: $F(x) = x \exp(-x^2/2)$
- Logistic: $F(x)=(1+e^{-x})^{-1}$

The activation function limits the node's response. No matter how weak or strong the inputs, the activation function limit the response to -1 to +1. This is a crucial feature of neural networks and part of the reason why a neural network deals with outliers so well [10].

As shown in Figure 1, neural networks have nodes arranged in a series of layers. The first layer is called the input layer, and the last one is known as the output layer. The number of nodes in the input layer corresponds to the number of independent variables, and the number of nodes in the output layer corresponds to the number of dependent variables. The layers of nodes in between the input and output layers are called hidden layers.

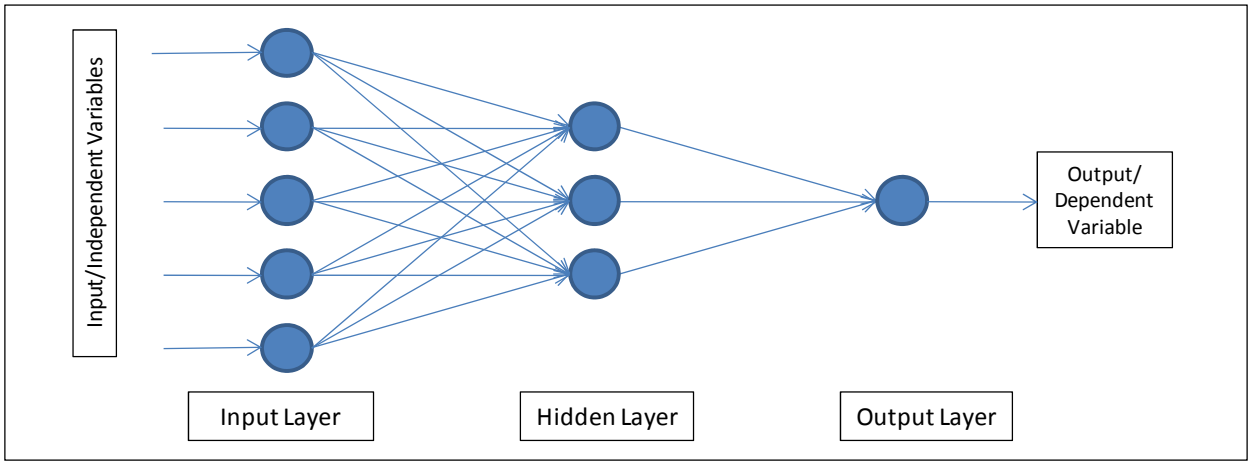

Figure 1: Neural Network Computational Structure

The commonly used neural networks are feed-forward neural networks, in which the only permissible connections among nodes are directed downward. Connections are not allowed among nodes of the same layer or the proceeding layer. If a neural network has hidden layers, which use nonlinear activation functions such as the hyperbolic tangent form, the model becomes genuinely nonlinear in its parameters, and the resulting model is known as a multilayer perceptron.

The neural network is trained by repeatedly giving it examples from its training set. A training set consists of an input vector paired with a corresponding output vector. Each example from the training

set is offered to the neural network and its output is calculated. The error between the calculated result and the real result is used to modify the matrix of weights between each layer in the network. The error adjustment is termed back propagation, which is a conceptually simple iterative gradient descent algorithm [11].

In training neural networks, the following three training parameters are required: learning rate, momentum, and training tolerance. Learning rate limits or multiplies the extent of weight changes in any given iteration. A high learning rate that reacts very quickly to input changes tends to make the network unstable. However, if the learning rate is lower than optimum, the neural network will take a substantially longer time to learn. A consensus among researchers is that adaptive learning rates are preferred as they can stabilize and accelerate convergence to a desired solution [12]. The momentum factor determines the proportion of the last weight change that is added to the new weight change. Low momentum often causes oscillation of weights and renders the network unstable, and learning is never completed. High momentum corresponds to a lack of flexibility and adaptability on the part of the neural network. In general, the momentum factor should be less than one (unity) to stabilize backpropagation [13]. The training tolerance factor specifies the margin of error allowable when neural network outputs are compared to real outputs. A training tolerance of zero indicates that the neural network outputs must exactly match the real outputs. A training tolerance close to zero can adversely affect the ability of the model to generalize, as a high degree of accuracy in the model is desired relative to training data. However, a high training tolerance factor is also not recommended, as it will result in inaccurate results because the specified accuracy is low [14]. In summary, balance must be achieved in specifying the training parameters for a neural network. The training parameters are application specific and are usually determined by trial and error.

Compared with conventional statistical procedures, neural networks usually use more parameters, and are thus more susceptible to overtraining. The overtraining phenomenon is observed when the mean squared error of the neural network continues to increase while the network performance is still improving in learning the training set. This is highly undesirable as it signifies that the neural network cannot recognize unknown patterns and its generalization ability is hampered [15]. A commonly used method in dealing with the overtraining phenomenon is to divide the data into a training set and a test set. The training set is used to fit the neural network model, and the test set is used to evaluate the model's performance.

4. Neural Network Simulation Meta-model

In order to explore the possibility of fitting a neural network with the outputs of the DES model, the author chooses three inputs and one output of the DES model. The three chosen inputs are: 1) patient arrival rate; 2) average DI/lab test time; and 3) the average boarding time, which is the average waiting time for inpatient bed after a patient is admitted. The chosen output from the DES simulation is the average waiting time for 1st EP assessment, which is the duration between the time when a patient arrives in the ED and the time when the patient is first seen by an EP. This waiting time does not include the time the EP spent with the patient.

For simplicity purpose, the three inputs are represented as percentage of their values used in the baseline DES model, which represents the current operation of the FMC ED. For example, if the patient arrival rate is 110%, this means that the patient arrival rate in the baseline DES model will be increased by 10%. The output, which is the average waiting time for 1st EP assessment, uses it true value.

To construct the training set for the neural network, the DES model is run at four values for each of the three inputs:

1) Patient arrival rate: 100%, 110%, 120%, 130%

2) Average DI or lab test time: 100%, 90%, 80%, 70%

3) Average boarding time: 100%, 90%, 80%, 70%

This generates a total of $4^3 = 64$ possible input configurations. For each configuration, the DES model is run for 10 times, each with a replication length of 485 days including 120 days as the warm-up period. From each set of ten results, the average and standard deviation of the waiting time for 1st EP assessment are calculated. The 99% confidence interval (CI) and its lower and upper bounds for the waiting time for 1st EP assessment are also calculated. The lower and upper bounds of the 99% CI from each input configuration are made available to the neural network and become its training set. The reason for using both the lower and upper bounds of the 99% CI is to retain the stochastic variability of the DES simulation.

For the generation of the test set for the neural network, the DES model is run at another three values for each of the three inputs:

1) Patient arrival rate: 105%, 115%, 125%

2) Average DI or lab test time: 95%, 85%, 75%

3) Average boarding time: 95%, 85%, 75%

This generates another total of $3^3 = 27$ input configurations to generate lower and upper bounds of the 99% CI for the waiting time for 1st EP assessment.

The configuration of the chosen neural network is shown below:

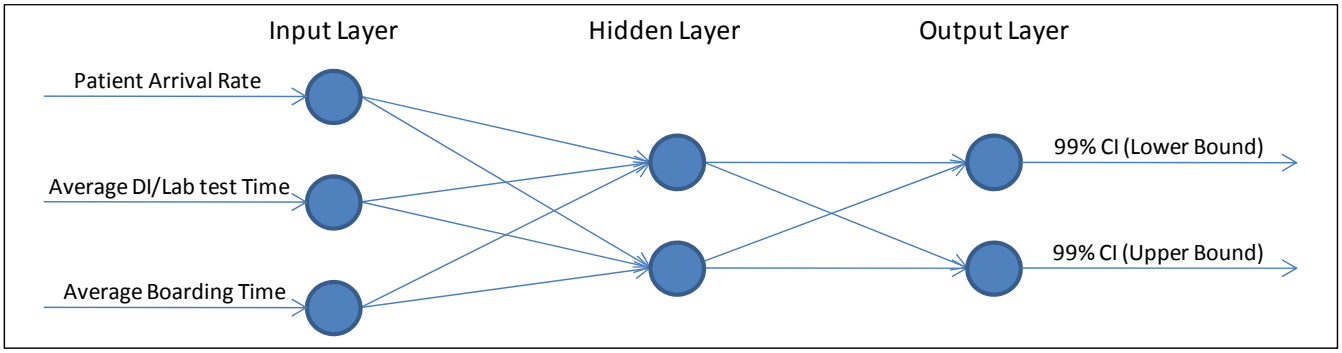

Figure 2: The three-layer feed-forward neural network model

The neural network has a three-layer feed-forward structure consisting of:

1) Input layer: 3 nodes
2) Hidden layer: 2 nodes
3) Output layer: 2 nodes

The Multilayer Perceptron (MLP) Module of IBM SPSS Statistics 17 is used for the development and test of the neural network model. The major MLP configurations are summarized below:

- Activation function: Hyperbolic tangent for hidden layer; and identity for output layer
- Rescaling of Scale dependent variables: standardized to be in the range (0-1)
- Type of Training: Batch mode
- Optimization Algorithm: Gradient descent
- Training Options:

  1) Initial learning rate: 0.4;

  2) Momentum: 0.9;

  3) Interval center: 0; and

  4) Interval offset: +/-0.5

- Stopping Rules

  1) Maximum steps without a decrease in error: 100,000,000

2) Maximum training time: 2 hours

3) Maximum training epochs: 100,000,000

4) Minimum relative change in training error: 1.0e-12

5) Minimum relative change in training error ratio: 1.0e-12

After being trained, the neural network model is used to predict equivalent simulation results for the 27 input configurations in the test set. For each input configuration, the neural network will predict the lower and upper bounds of the 99% CI for the waiting time for 1st EP assessment. The predicted average waiting time for 1st EP assessment will be calculated by averaging the lower and upper bounds of the 99% CI. Then the percent difference between simulated average and predicted average is calculated using the following formula:

Let:

MT_{sim} = Average waiting time for 1st EP Assessment simulated using the DES model

MT_{NN} = Average waiting time for 1st EP Assessment predicted by the neural network

PER_{Dif} = Percent difference between the average waiting time for 1st EP Assessment simulated using the DES model and the average waiting time for 1st EP Assessment predicted by the neural network

Then:

$$PER_{Dif} = (MT_{NN} - MT_{sim}) / MT_{sim} \quad (1)$$

The results are shown in Figure 3. In the Figure, X-axis represents the 27 input configurations in the test set. The left Y-axis represents the average waiting time for 1st ED assessment in minutes. The right Y-axis represents the percent difference between simulated average and predicted average (PER_{Dif}). The values simulated using the DES model **a**re represented with the dotted line, and those predicted by the neural network are represented with diamond shapes. The values of the PER_{Dif} are represented with circle shapes.

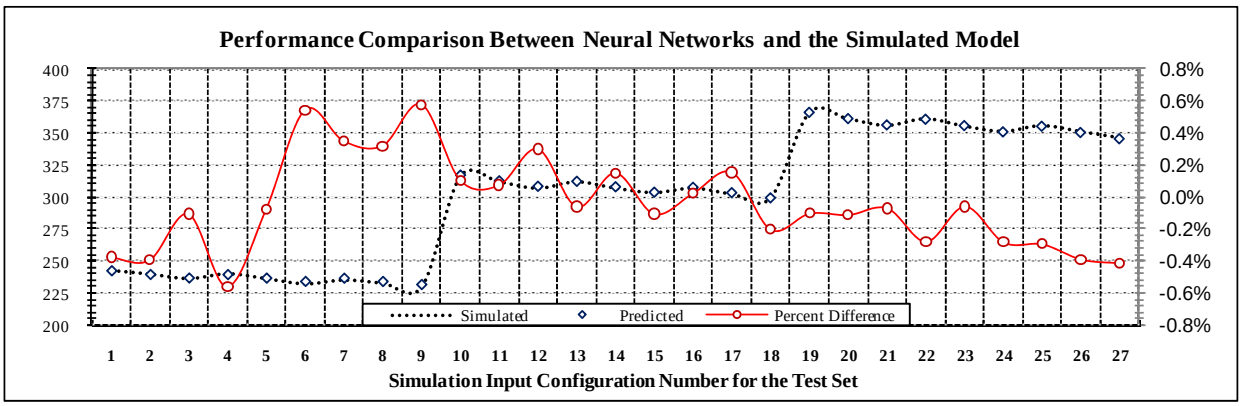

Figure 3: Comparison between the DES model and Neural Network model for the Test Set

The result shows that the values predicted by the neural networks are within +/- 0.6% of the values simulated using the DES model.

In addition, we also used the same training set to develop multiple regression (MR) models for the 99% CI lower bound and upper bound in SPSS. The summary of the two MR models is shown below:

Table 2: Summary of the two Multiple Regression models

Model	R		R Square	Adjusted R Square		
	Training Set	Test Set			F	Sig.
99% CI Lower Bound	.986	.993	.971	.970	679.117	.000
99% CI Upper Bound	.986	.995	.971	.970	675.185	.000

The two MR models are used to predict the lower and upper bounds of the 99% CI for the waiting time for 1st EP assessment, and the predicted average waiting time for 1st EP assessment will be calculated by averaging the predicted lower and upper bounds of the 99% CI.

The results from the simulation model, the neural network model and the multiple regression models are shown in Figure 4. The values simulated using the DES model are represented with the dotted line, and those predicted by the neural network are represented with diamond shapes. The values predicted by the MR models are represented with triangle shapes.

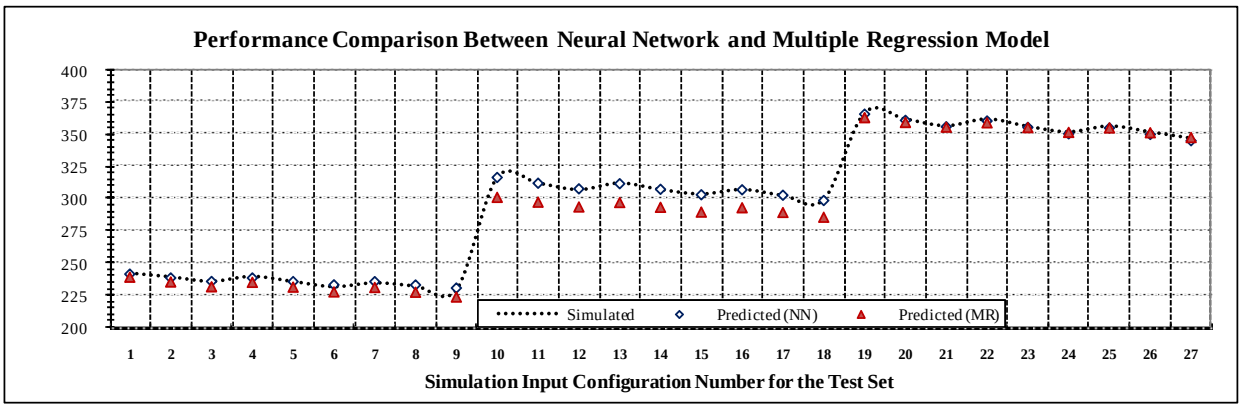

Figure 4: Comparison between the DES, neural and multiple regression models for the Test Set

The result shows that the neural network model outperforms the multiple regression models at almost all the data points.

5. Different Neural Network Configurations

In this section, we intend to investigate the impact of the neural network configuration on its performance. The chosen neural network configurations include the following two options:

Option 1: using only one hidden layer

Option 2: using two hidden layers with equal number of nodes in each hidden layer

For option 1, the number of nodes in the hidden layer ranges from 2 to 9. For option 2, the number of nodes in the hidden layers ranges from 2 to 6.

Each neural network is trained using the same training set. Then it is used to predict the values for input configurations in both the training set and the test set. For each input configuration, the percent difference between simulated average and predicted average waiting time for 1st ED assessment is calculated using formula (1).

Figure 5 shows the PER_{Dif} values for different neural network configurations calculated for the training set and the test set. The X-axis in Figure 5 represents the configurations of the neural networks, and the Y-axis represents the percent difference between the simulated average and the predicted average

waiting time for 1st EP assessment. The solid line represents the result for the training set, and the dotted line represents the result for the test set.

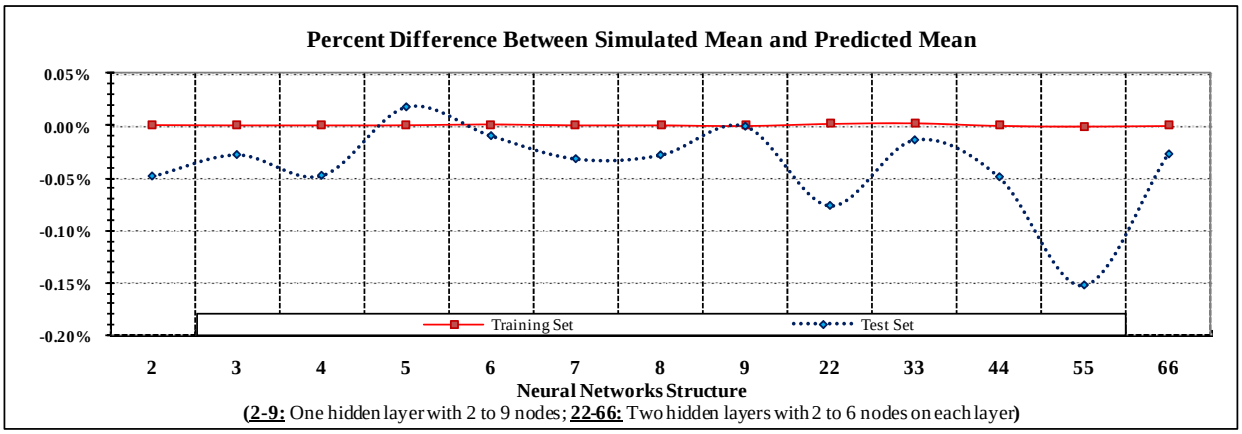

Figure 5: Percent Difference between Simulated Average and Predicted Average

From Figure 5, we can see that all the neural networks can very accurately reproduce the average waiting times for 1st EP assessment for the training set with PER_{Dif} close to zero. For the test set, the values of the PER_{Dif} vary between -0.15% and 0.02%, and neural networks with the following configurations have relatively better performance compared with the neural networks with other configurations:

1) Option 1: 3/6/9 nodes in the hidden layer
2) Option 2: 3/6 nodes in each hidden layer

Another close look at these neural networks with better performance shows that these networks have one thing in common:

The number of nodes in the hidden layers equal to the integer times of the number of the input variables

If we compare the performance of the following configurations for option 1: 3 nodes, 6 node and 9 nodes in the hidden layer. The performance of the neural network improves with the increase of the number of the nodes. However, if we compare the following two configurations for option 2: 3 nodes

and 6 nodes, the performance of the 6-node configuration is worse than the 3-node configuration. One possible explanation is that the neural network with 6 nodes in hidden layers may have been overtrained by the training set.

6. Discussion and Conclusions

Neural networks tend to perform better than traditional multiple regression models in modeling non-linear input-output relationships. This makes neural networks ideal candidates for simulation meta-models as the relationships between the inputs and outputs of most simulation models are non-linear (Otherwise, there should be no need to use computer simulation). In this paper, a simple neural network with one hidden layer consisting of two nodes was used to model an ED DES simulation with three chosen inputs and one output. After being trained, the neural network model can accurately predict the outputs of the ED simulation for inputs in the test set, which are not included in the training set. In the case presented in this paper, the trained neural network also outperforms the multiple regression models developed from the same training set. This is very encouraging.

The prediction of a neural network is deterministic, and the stochastic variability of the simulation will be lost. In order to overcome this shortcoming of the neural network model, two nodes were used in the output layer of the neural network model: one represents the lower bound of the 99% CI, and the other represents the upper bound of the 99% CI. With this design, the neural network can reproduce the average waiting time for 1st EP assessment and its 99% CI so that the stochastic variability of the simulation can still be retained.

One of the primary considerations when applying neural networks is the neural network configuration: number of hidden layers and number of nodes in each hidden layer. In the study reported here, the author test different neural network configurations. Analysis of the results shows that neural network with more nodes do not always lead to better performance. A further analysis shows that neural

networks with number of nodes in the hidden layers equal to integer times of the number of input variables tend to perform better than neural networks with other configurations. If we only compare the performances of these neural networks, it shows that neural networks with more nodes tend to perform better than those with fewer nodes. However, there are exceptions. Some neural networks with more nodes perform worse than those with fewer nodes for the test set though they perform better for the training set. The author thinks this may be because these neural networks have been overtrained by the training set. In the analysis of the results, the author also finds out that neural networks with more nodes take longer to be trained than the neural networks with fewer nodes.

Choosing the appropriate training rules is another very time-consuming trial and error process. Balance need to be made between undertraining and overtraining. It will become even more complex when the objective is to compare the performance of neural networks with different number of nodes. Using the same rules may lead to the overtraining of the neural network with fewer nodes and/or undertraining of the neural network with more nodes. Adopting different rules for different neural networks may have the risk of undertraining the neural networks with fewer nodes. In general, the choice of training rules should be dealt with on case by case basis rather than trying to find a set of common rules applying to everywhere.

In summary, this paper has shown that it is possible to train a neural network to model the generalized response of parameter changes in an emergency department DES simulation. With proper training, the neural network can accurately predict the simulation outputs for any inputs within the parameter domain included in the training set. This result has the potential to broaden and enhance the applications of simulation modeling techniques for healthcare management. Some possible applications include: 1) used as an interaction within other simulation models or software systems to offer its immediate view of changes in the original simulation models without the requirement of rerunning the simulation model; 2)

used as a tool to perform response surface analysis and optimization; 3) used as a tool to combine the powers of different simulation models without the need to physically link these simulation models together.

But we should also be aware that there are limitations for a neural network model. First, it is only valid for the specified parameter domain included in the training set. Second, changes to the simulation (e.g. distribution parameters, addition or subtraction of variables) would invalidate the neural network meta-model, and a new meta-model would be needed to reflect the altered simulation. Third, there is a lack of commonly accepted rules for determining the optimal number of nodes to put in a hidden layer. Trial and error methods are necessary, especially if the functional input-output relationships are not known. Most of the times, this can be a very tedious and time consuming process. Forth, special attentions need to be paid to the rules for training the neural network. Using stringent rules can lead to overtraining, and using loose rules can lead to undertraining. Trial and error methods are also necessary to reach a reasonable balance.

Future research efforts in this area should include how to design and choose training and test sets, neural network configuration and training rules to optimize the performance of the trained neural network. Other research efforts should include the exploration of using trained neural network in various applications for healthcare management.

Acknowledgements

The author thanks Dr. Paul Rogers, Department of Mechanical and Manufacturing Engineering, Schulich School of Engineering, University of Calgary and Dr. Thomas Rohleder, Health Care Policy and Research, Mayo Clinic for their support for this research.

References

1. Valinsky, D. (1975). Simulation. In L. J. Shuman, R. D. Speas Jr., & J. P. Young (Eds.), Operations Research in Health Care: A Critical Analysis. Baltimore: Johns Hopkins University Press, 114-176.
2. Jun, J. B., Jacobson, S. H., and Swisher, J. R. (1999). Application of discrete-event simulation in health care clinics: a survey. Journal of the Operational Research Society, 50(2), 109-123.
3. Jacobson, S. H., Hall, S. N., and Swisher, J. R. (2006). Discrete-Event Simulation of Health Care Systems. In R. W. Hall (Ed.), Patient Flow: Reducing Delay in Healthcare Delivery: Springer, 211-252.
4. Brailsford, S. 2005. Overcoming the barriers to implementation of operation research simulation modeling in healthcare. Clinical and investigative medicine. Medicine Clinique et experimentale, 28:312-315.
5. Proudlove, N.C., Black, S. and Fletcher, A. 2007. OR and the challenge to improve the NHS: modeling for insight and improvement in –patient flows, Journal of Operational research Society 58:145-158.
6. Yu, B., and K. Popplewell, 1994, "Metamodels in manufacturing: a review," International Journal of Production Research, vol.32, 787-796.
7. Mukesh, D. (1997, March). Hate Statistics? Try neural networks. Chemical Engineering, pp. 96-104.
8. Stern, H.S. (1996), Neural networks in applied statistics, Technometrics, 38, 205-214
9. Sarle, W.S. (1994), Neural networks and statistical models. Proceedings of the SAS Users Group International Conference, pp. 1528-1550

10. Detienne, K.B., DeTienne, D.H., Joshi, S.A., Neural networks as statistical tools for business researchers, Organizational Research Methods, Apr 2003; 6,2; ABI/INFORM Global, pp. 236.
11. Rumelhart, D.E., Hinton, G.E., & William, R.J. (1986). Learning internal representations by error propagation. In D.E. Rumelhart & J.L. McClelland (Eds.), Parallel distributed processing: Explorations in the microstructure of cognition (pp. 318-362). Cambridge, MA: MIT Press.
12. Looney, C.G. (1996). Advance in feed-forward neural networks: Demystifying knowledge acquiring black boxes. IEEE Transaction on Knowledge and Data Engineering, 8, 211-226.
13. Yu, X.H., & Chen, G.A.(1997). Efficient backpropagation learning using optimal learning rate and momentum. Neural Networks, 10, 517-527.
14. Kuo, C., & Reitsch, A., (1995). Neural networks vs. conventional methods of forecasting. Journal of Business Forecasting Methods and Systems, 14(4), 17-22.
15. Tzafestas, S.G., Dalianis, P.J., & Anthopoulos, G. (1996). On the overtraining phenomenon of backpropagation neural networks. Mathematics and Computers in Simulation, 40, 507-521.